# APPENDIX A

## Table of Contents



## Figures

## Executive Summary

ABC World Industries (ABC) is a leading manufacturer of flooring and ceiling products and has been an Services Corporation customer for more than two years. Over time ABC has gone through a number of changes, both technical and organizational, which has prompted a re-evaluation of the network design that was implemented over two years ago.

ABC as an organization is broken up into different business units based on the product that is manufactured.

ABC made another decision to divest itself of its building insulation products, which was called ABC Industrial Products (AIP), which resulted in a separate, autonomous company called XYZ.

The purpose of this document is twofold. The first is to profile the performance characteristics of the accounts payable and accounts receivable components of the SAP R/3 Financials module. The current SAP R/2 system based in Europe will be phased out and all of the SAP R/2 users will migrate over time to the SAP R/3 instance located in USA. Although this transition has not started yet this document will set forth the bandwidth requirements of those components using simulation and network modeling techniques.

The second goal of this document is to present the network re-design of each of the three business entities based on several planning sessions that have transpired during the fall of 2000.

Moving forward this document should be used as a baseline for making network design decisions. It is anticipated that as more information becomes available and more changes are implemented this document will need to be updated to reflect those changes.

This report documents 's findings. All of the methodologies and processes were carried out in accordance with 's Network Analysis Program.

The following three sections provide an overview of computing architectures in general and the processes and methodologies that uses as part of the Network Analysis Program.

**Computing Architecture**
Deployment of applications across a network requires careful planning and an understanding of several key aspects, such as an application's performance characteristics and computing architecture that it operates on.

Applications generally fall into one of five categories.

**Terminal/Host** - The flow of traffic is usually asymmetric. The terminal sends a few characters and the host returns many characters. Telnet is an example of an application that generates terminal/host traffic. The default behavior for Telnet is that the terminal sends a single packet for each character a user types. The host returns multiple characters, depending on what the user typed.

**Client/Server** - Client/server is the best known and most widely used traffic type. Examples of client/server implementations include NetWare, AppleShare, Banyan, Network File System (NFS), and Windows NT. The flow of traffic is usually bi-directional and asymmetric. Requests from the client are usually less than 64 bytes, except when writing to the server, in which case they are larger. Responses from the server range from 64 bytes to 1500 bytes or more, depending on the maximum frame size allowed for by the data-link layer in use.

**Peer-to-Peer** - The flow of traffic is usually bi-directional and symmetric. Communicating entities transmit approximately equal amounts of protocol and application information and typically there is no hierarchy. Each device is considered as important as each other device, and no device stores substantially more data than any other device.

**Server/Server** - Server/server traffic includes transmissions between servers and transmissions between servers and management applications. Servers talk to other servers to implement directory services, to cache heavily-used data, to mirror data for load balancing and redundancy, to back up data, and to broadcast service availability. Servers talk to management applications for some of the same reasons, but also to enforce security policies and to update network management data. The flow of traffic is generally bi-directional and the symmetry of the flow depends on the application. With most server/server applications, the flow is symmetrical, but in some cases there is a hierarchy of servers, with some servers sending and storing more data than others.

**Distributed Computing** - Distributed computing refers to applications that require multiple computing nodes working together to complete a job. Characterizing traffic flows for distributed computing applications most likely will require that the data is studied using a protocol analyzer and/or modeled using a network simulator.

Today's enterprise applications evolved from host/terminal systems. With the explosion of the Internet and the World Wide Web, systems have been transformed into Internet-enabled, multi-tiered applications.

In a three-tier architectural environment the user interface (client), the business logic layer (application server) and the database layer (database server) are separated into three distinct components. Each component can have one or more functions. For example, there can be one or more user interfaces in the top tier, and each user interface may communicate with more than one application in the middle tier at the same time. Applications in the middle tier may use more than one database at a time. Components in a tier may run on a computer that is separate from the other tiers, communicating with the other components over a network. In a two-tier architecture the user interface and business logic components are combined into a single layer while the database layer remains separate. The development and advancement of fourth-generation languages (4GL) have helped popularize this approach.

Vendors have moved to three-tier applications primarily to increase scalability. Three-tier architecture reduces network traffic between the client and the application server, which allows more users to operate on the network, and also improves application response time in many instances. In addition multiple servers can be deployed at the mid-tier, which enables the transaction load to be balanced across multiple servers. The following diagram illustrates a simple three-tier architecture.

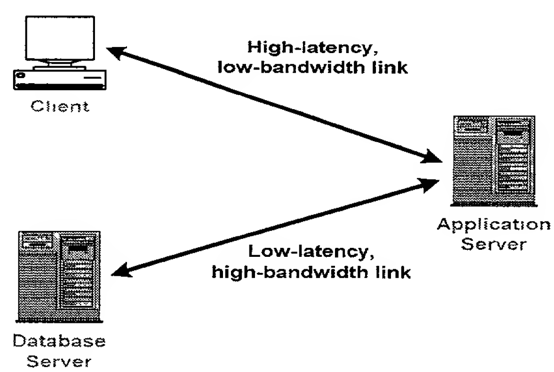


**Figure 1: Three-Tier Architecture**

In this scheme, thin clients provide users access to the application server to process the business logic; multiple servers can be added to improve scalability. Three-tier applications require a high-bandwidth, low-latency link between the application server and the database server because of the volume of data transferred between the two. The client-to-application server link does not have to be as fast.

It is important to note that not all vendors have created equivalent implementations of multi-tier architectures. Enterprise Resource Planning (ERP) applications from Oracle and SAP are both based on three-tier architectures. However, each one varies in the amount of network traffic it generates. Oracle sends many small messages between the client and the application server, which reduces bandwidth usage but increases susceptibility to network latency. SAP sends large messages, which increases bandwidth usage but reduces susceptibility to network latency.

Many of the ERP vendors provide JAVA-based, Web-enabled clients as well as their proprietary graphical user interface (GUI). Web-based enterprise applications differ from three-tier applications in that they use the standard HTTP protocol for some of the communications between the client and mid-tier, which requires a Web server and potentially an additional layer. The following diagram illustrates the various components.

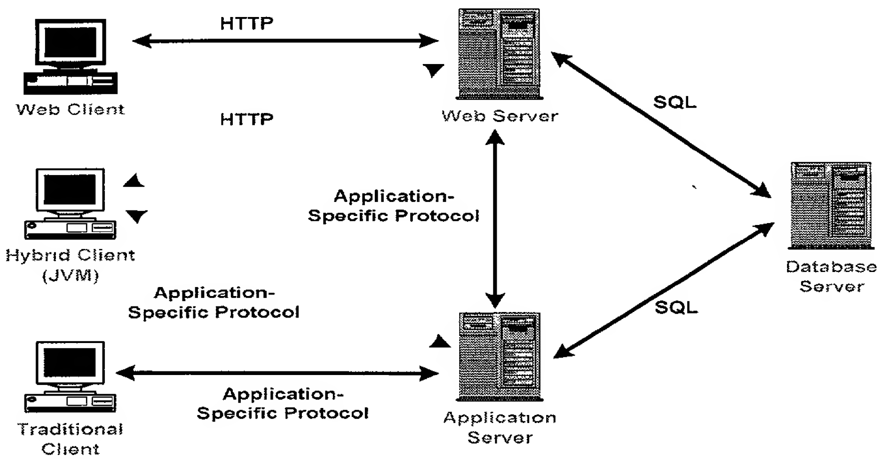


**Figure 2: Enterprise Application Network Architecture**

Three-tier applications use an application-specific protocol to communicate between the client and the mid-tier. Web-based ERP clients follow a hybrid model. HTTP is used to download a JAVA applet containing the client code, which is executed at the client by a JAVA Virtual Machine (JVM). The JAVA applet then uses the application-specific protocol to communicate with the mid-tier application server. This model uses a Web browser to provide a common user interface, but does not use HTTP for all of the client-to-mid-tier communications. The Web server in an Internet-based application may communicate with the database server directly or indirectly via the application used by traditional clients.

**Profiling Methodology**

To understand the performance characteristics of an application it is necessary to analyze the data that is generated by all of the associated functional tasks and components, which then serve as a baseline. The key to creating an accurate baseline is to eliminate factors that could adversely affect an application's performance such as router or switch transit delays, multiple router hops, network latency and congestion. Consequently a baseline should not be done across a WAN. Although a local area network (LAN) introduces some degree of network latency and congestion, it does provide a good environment for creating a baseline on how an application truly behaves. This data can be used to do predictive analysis on how an application would perform across a WAN with varying amounts of network latency and bandwidth. This process is also referred to as *application profiling* or *traffic mapping.*

The key is to develop an overall plan detailing usage patterns and traffic priorities. Understand the number and distribution of users, their usage patterns, and the business priorities of the applications. The goal is to determine how much data is generated in each direction along with the complexity of the underlying communication. The complexity of communication is measured in terms of *turns*, which can take place at both the protocol and application level. A turn is defined as a complete request/response transaction or sequence of packets that is either initiated by the protocol or application. Because there is an associated amount of latency with every *turn*, as the number of *turns* increases there may be degradation in response time.

The testing process involves a user executing application tasks from a client located on the same LAN segment as the server, and ideally on the same broadcast domain. While the user runs the application, the data is captured. The tasks should represent typical transactions performed by a user on a daily basis. Although any given application typically contains hundreds, if not thousands of commands and functions, experience shows that an application generally exhibits common communication characteristics across a common set of functional tasks. Consequently, it is not necessary to test every aspect of an application to understand how it performs. The data collection process many times has to be done several times during the day or month in order to account for fluctuations in applications usage that can be associated with month-end processing or product rollouts.

The *application profiling* process focuses on several application performance characteristics. The first is application efficiency. Efficiency refers to whether an application's use of the underlying protocol uses bandwidth effectively. Efficiency is affected by frame and packet size, the interaction of protocols used by the application, windowing and flow control, and error-recovery mechanisms. The key to efficiency is to minimize the amount of protocol overhead in order to maximize the amount of payload information (application data) within each packet.

Data symmetry is the comparison between the amount of data that is generated by the client as part of a request and the amount of data that is sent back from the server as part of the response to the original request. Data symmetry is also referred to as the ratio of server to client data. The importance of understanding the data symmetry is that it can help in making cost effective and efficient network design decisions.

Response time is the one of the most important aspects of an application. Regardless of whether an application is "bursty" or whether it communicates efficiently, the bottom line is that when users run the application they expect a certain level of performance. It is important to understand what network components have the greatest impact on an application's response time; this aspect is also called application sensitivity. The goal is to determine whether an application is more sensitive to varying amounts of bandwidth or network latency.

By combining the knowledge of an application's protocol efficiency, data symmetry and sensitivity along with the knowledge of an underlying network infrastructure, an effective solution can be designed which takes advantage of the application's strengths and avoids its weaknesses.

**Network Modeling**

One of the most difficult tasks of network design is to accurately predict how applications will perform under certain conditions. Over the years tools have been developed which help network designers make informed decisions. The gamut of tools available on the market today range from very bad to very good. Regardless of the quality, to use any tool effectively requires an in-depth understanding of network design principles and application architecture.

Network modeling can be done in one of two ways. The first approach is to use analytical methods. Using mathematical algorithms, estimates can be made of link (or virtual circuits) utilization or network latency. Unfortunately there are some deficiencies with this method. Protocol effects are difficult to capture. Important protocol aspects that are extremely difficult to

represent in a mathematical network model include data segmentation, congestion control, retransmissions, load balancing across multiple routes, and sophisticated algorithms such as selective acknowledgements in TCP, or weighted fair queuing in IP.

Another approach is to use discrete event simulation methods. By either manually building the unique characteristics of a network and its various components, or drawing upon a library of predefined components, it is possible to generate explicit network traffic and create an accurate baseline. Once this baseline is created, multiple "what if" network design scenarios can be simulated in order to measure a multitude of network and application metrics such as application response time, link utilizations, and throughput. These scenarios can include an increase in the user population over time or the addition of new applications. The advantage of using discrete event simulation methods is that the model can accurately reflect the uniqueness and nuances of a specific application and/or network.

uses a modeling tool that uses a hybrid approach, which combines both analytical and discrete event simulation methods. The advantage of a hybrid approach is that accurate results can be obtained quickly and efficiently.

An important aspect of the application analysis process is the ability to accurately size the connection (network access circuit and network access port) between end-users (remote site) and the application and database servers (host site). To help in this process has developed a proprietary modeling tool called the *ROI Solution Builder*, which calculates the minimum size of the network access circuit needed to support a given number of users. The modeling tool uses very detailed application information that is collected during the testing period. In addition the *ROI Solution Builder* also provides the ability to compare two design scenarios against one another in order to calculate a financial return on investment.

**Test Environment**

The data collection portion of the analysis took place at ABC's USA facility. The testing consisted of using multiple clients accessing the SAP R/3 system.

- Date — October<br>USA
- Client — Microsoft Windows 95<br>SAPGUI v4.5B
- Application Server — HP 9000 Series 800<br>HP-UX 11.0<br>SAP v4.5B
- Database Server — HP-UX 11.0<br>Oracle v8.05
- Network — Switched 100Base-T
- Protocol — TCP/IP

The following diagram is a graphical view of the test environment. Although the actual network consists of many more components and there are a total of ten HP 9000 SAP servers, this diagram is the network simulation model that was created in order to baseline the SAP application. During simulation it is not necessary to recreate the production environment exactly.

**Figure 3: Test Environment – USA**

## Test Results

The tables on the following pages represent all of the application tasks that were captured during the testing. Many of the statements made in regards to the performance characteristics of a particular application are based on the data from these tables. The following is a description of the fields that appear in the tables.

| | |
|---|---|
| • Total Bytes | Total bytes transmitted. |
| • App Turns | Total application turns. |
| • Bytes/App Turn | Calculation based on Total Bytes / App Turns. |
| • Client Bytes – Total | Total bytes sent from the client to the server. |
| • Client Bytes – Payload | Total amount of application data sent from the client to the server. |
| • Client Bytes – Overhead | Percentage of protocol data sent from the client to the server. |
| • Server Bytes – Total | Total bytes sent from the server to the client. |
| • Server Bytes – Payload | Total amount of application data sent from the server to the client. |
| • Server Bytes – Overhead | Percentage of protocol data sent from the server to the client. |
| • Ratio | Relationship between Server Total Bytes and Client Total Bytes. |
| • Duration | Total amount of elapsed time (in seconds). |
| • Overall Protocol Overhead | Percentage of protocol data transmitted as part of the entire application. |
| • Overall Server to Client Ratio | Relationship between the sum of all Server Total Bytes and the sum of all Client Total Bytes. |
| • Average Client Transaction | Geometric mean of Client Total Bytes based on all individual tasks. |
| • Average Server Transaction | Geometric mean of Server Total Bytes based on all individual tasks. |
| • RV | Specifies the Report Version |

## SAP - Accounts Payable

(RV 8.5)

| | Total Bytes | App Turns | Bytes/App Turn | Client Bytes Total | Client Bytes Payload | Client Bytes Overhead | Server Bytes Total | Server Bytes Payload | Server Bytes Overhead | Ratio | Duration |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Prepare PO Invoice | 14047 | 9 | 1561 | 1982 | 794 | 59.9% | 12065 | 10943 | 9.3% | 6:1 | 46.72 |
| Prepare PO Invoice | 6487 | 4 | 1622 | 894 | 366 | 59.1% | 5593 | 4999 | 10.6% | 6:1 | 11.83 |
| Prepare PO Invoice | 9007 | 5 | 1801 | 1126 | 466 | 58.6% | 7881 | 7155 | 9.2% | 7:1 | 15.54 |
| Prepare PO with Planned Freight | 20520 | 16 | 1283 | 4312 | 2332 | 45.9% | 16208 | 14162 | 12.6% | 4:1 | 49.29 |
| Prepare PO with Planned Freight | 26524 | 18 | 1474 | 4718 | 2474 | 47.6% | 21806 | 19628 | 10.0% | 5:1 | 42.80 |
| Prepare PO with Planned Freight | 26822 | 18 | 1490 | 4698 | 2454 | 47.8% | 22124 | 19880 | 10.1% | 5:1 | 45.27 |
| Prepare Check Request | 16398 | 11 | 1491 | 3357 | 2103 | 37.4% | 13041 | 11523 | 11.6% | 4:1 | 57.02 |
| Prepare Check Request | 16324 | 11 | 1484 | 3322 | 2068 | 37.7% | 13002 | 11484 | 11.7% | 4:1 | 39.80 |
| Prepare Check Request | 19778 | 14 | 1413 | 3973 | 2323 | 41.5% | 15805 | 14155 | 10.4% | 4:1 | 56.61 |
| Manually Approved Invoice | 16548 | 11 | 1504 | 3392 | 2072 | 38.9% | 13156 | 11704 | 11.0% | 4:1 | 55.58 |
| Manually Approved Invoice | 17019 | 11 | 1547 | 3377 | 2057 | 39.1% | 13642 | 12190 | 10.6% | 4:1 | 43.51 |
| Manually Approved Invoice | 16236 | 11 | 1476 | 3291 | 2037 | 38.1% | 12945 | 11427 | 11.7% | 4:1 | 44.79 |
| Freight with Accrual | 17211 | 11 | 1565 | 3438 | 2118 | 38.4% | 13773 | 12453 | 9.6% | 4:1 | 76.67 |
| Freight with Accrual | 25815 | 17 | 1519 | 4673 | 2561 | 45.2% | 21142 | 19228 | 9.1% | 5:1 | 65.06 |
| Freight with Accrual | 25837 | 17 | 1520 | 4687 | 2575 | 45.1% | 21150 | 19236 | 9.0% | 5:1 | 58.27 |
| Freight with Accrual | 15967 | 10 | 1597 | 3185 | 1997 | 37.3% | 12782 | 11462 | 10.3% | 4:1 | 39.94 |

**Average client transaction is 3.1KB**
**Average server transaction is 13.9KB**

**Overall protocol overhead is 16.6%**
**Overall server to client ratio is 4:1**

**Figure 4: Accounts Payable - Test Results**

## SAP - Accounts Receivable

(RV 8.5)

| | Total Bytes | App Turns | Bytes/App Turn | Client Bytes Total | Client Bytes Payload | Client Bytes Overhead | Server Bytes Total | Server Bytes Payload | Server Bytes Overhead | Ratio | Duration |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Posting Wire Transfer | 57877 | 35 | 1654 | 7840 | 3154 | 59.8% | 50037 | 46407 | 7.3% | 6:1 | 204.18 |
| Lock Box Entry - Activation | 24001 | 14 | 1714 | 2876 | 1160 | 59.7% | 21125 | 19277 | 8.7% | 7:1 | 52.04 |
| Lock Box Entry - Activation | 52622 | 37 | 1422 | 7811 | 2927 | 62.5% | 44811 | 41577 | 7.2% | 6:1 | 101.68 |
| Lock Box Entry - Fast Entry | 48485 | 34 | 1426 | 7628 | 3338 | 56.2% | 40857 | 37623 | 7.9% | 5:1 | 463.23 |

**Average client transaction is 6.1KB**
**Average server transaction is 37.3KB**

**Overall protocol overhead is 15.0%**
**Overall server to client ratio is 6:1**

**Figure 5: Accounts Receivable - Test Results**

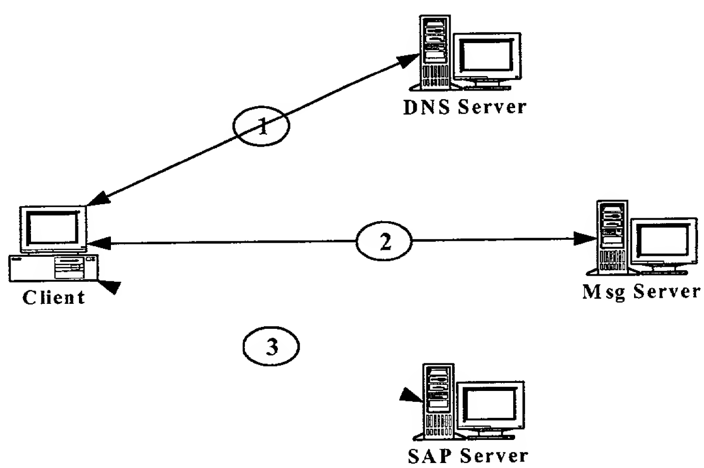


**Figure 6: SAP Login Process**

**SAP R/3**

Although ABC uses many of the SAP modules the testing for this analysis focused on the accounts payable and accounts receivable components of the Financials module. When the SAP R/2 system is phased out in Europe, users will be migrated over to the SAP R/3 instance in USA, which will have a significant impact on the network usage across the Frame Relay network.

**Background**

SAP AG (Walldorf, Germany) is a leading manufacturer of an enterprise resource planning software package called SAP R/3 (R/3). R/3 is extremely flexible and provides a company with the ability to integrate all of its business operations (planning, controlling and monitoring) into a central, integrated system. At its core are modules (programs) for accounting and controlling, production and materials management, quality management and plant management, sales and distribution, human resources management, and project management. One of the key advantages of R/3 is that it overcomes the limitations of traditional hierarchical and function-oriented systems by integrating all software components into a single workflow of business events and processes across departments and functional areas.

Designed as an open architecture, R/3 works seamlessly with a variety of systems and applications, which allows for many different options for useful add-on applications and cooperative information processing. The Business Framework, R/3's strategic product architecture, enhances this openness. Although designed as an integrated system, R/3's object-oriented interfaces allow specific business functions to operate as standalone software products, without any loss of integration. Workflow applications automate and control the flow of information, and transport documents such as orders or invoices from one work center to another. Workflow management speeds the flow of budget releases and purchase requisitions, increases the efficiency of change management in engineering/design and manufacturing, and simplifies subsequent processing of documents transmitted by fax or EDI.

R/3 supports a number of different types of clients, hardware platforms, operating systems, and databases and can be configured to operate as either a two-tier or three-tier system. R/3 also provides Internet Application Components, which can be used to provide an HTML-enabled front-end to R/3 applications. The Internet Application Components handle the interaction between the Internet and the R/3 application server and provide firewall transversal capability. The bulk of the transaction is processed at the application server. Transparent load balancing across multiple application servers is also supported; new application servers can be added without users having to reconfigure.

Client services can be deployed using either a standard Web browser (Microsoft Internet Explorer or Netscape Navigator) or using R/3's desktop application called SAPGUI (Windows 3.1, Windows 95/98/NT, OSF/Motif, OS/2 or Macintosh).

All communications between the client, application server, and database server use TCP/IP. Communication between the application server and database server is via SQL using RPC calls. Once an SQL connection is established it is maintained and reused for the duration of the session, which helps minimize the network traffic between the application server and database server by eliminating the necessity of having to constantly establish new TCP connections.

When evaluating an application from a performance perspective there are a number of aspects that need to be studied. One of the most important aspects is how the application has been designed and how that design impacts the network. One of SAP's strengths as a software vendor is that the R/3 system was designed to operate as an enterprise application. Although the initial design of R/3 was based on a two-tier architecture, SAP made a successful transition from a two-tier to a three-tier design, which helps in minimizing the traffic that is sent between client and the application server. One of the keys to R/3's efficiency is the way that it sends data from the client to the application server. As a client fills out a page (or screen), the information is sent to the application server in blocks, which increases bandwidth usage but reduces susceptibility to network latency.

**Efficiency**

Efficiency refers to whether an application uses the underlying bandwidth effectively. Efficiency is affected by frame size, the interaction of protocols used by the application, windowing, flow control, and error-recovery mechanisms. The key to efficiency is to minimize the amount of protocol overhead in order to maximize the amount of payload information (application data) within each packet. The testing revealed that the communications between the client and the application server are very efficient. There are two ways to make this determination.

The first way is to determine the amount of protocol overhead that is used during transmission and the second way is to determine the amount of data that is transferred per application turn. The concept of an application turn was introduced earlier in this document. Every application turn has an associated amount of latency, which means the more information that can be transferred per turn the faster and more efficiently the transaction will complete.

The testing revealed that the *overall* protocol overhead was fairly low and that the "Bytes/App Turn" was high for both accounts payable and accounts receivable. This information is displayed as part of the test results (figures 4 and 5). Another point is that the responses returned from the application server to the client contained much less protocol overhead as compared to the client requests that were made to the application server. This is ideal because although there is much more data flowing from the application server to the client (which can be expected), the transfer of that information is very efficient.

**Bandwidth Consumption**

Perhaps one of the most difficult aspects of designing a network solution is to determine how much bandwidth an application requires. There are two ways to address this issue. The first is to determine a minimum amount bandwidth that an application will need to operate on a per user basis, which is addressed later in this document. The second aspect is to determine how much the application bursts to consume bandwidth. This is an important aspect to understand because at times of *heavy* usage by many users, if an application tends to burst there needs to be enough bandwidth available on the circuit, irrespective of the underlying network, i.e., Frame Relay, IP, or ATM. If an application tends to perform using a fairly deterministic amount of bandwidth then it is easier to estimate the overall bandwidth requirements based on a given number of users.

Ideally the rate of transmission between various network devices should remain fairly constant. However, over time as applications have become more data intensive and graphically oriented, they have become more demanding. Depending on the task the rate of transmission between a

client and server could suddenly increase. The term that is typically used to describe this situation is bursty. The effect of bursty traffic is that an application could consume all of the available bandwidth at any given time depending on the task that is being executed.

Although R/3 is efficient in the way that it communicates the application tends to be burst, consuming available bandwidth. A contrast needs to be made between SAP and the way a program such as file transfer protocol (FTP) operates. Depending on the task, SAP will have a momentary burst in activity between the client and the application server, and once the task (or sub-task component) is complete the bandwidth requirement will subside. However, depending on the size of the file transfer, FTP will continue to increase its TCP window size based on the available amount of bandwidth, which will remain in effect for the duration of the file transfer.

The following diagram is an excellent example showing how the application has a lot of spikes during the posting of a wire transfer.

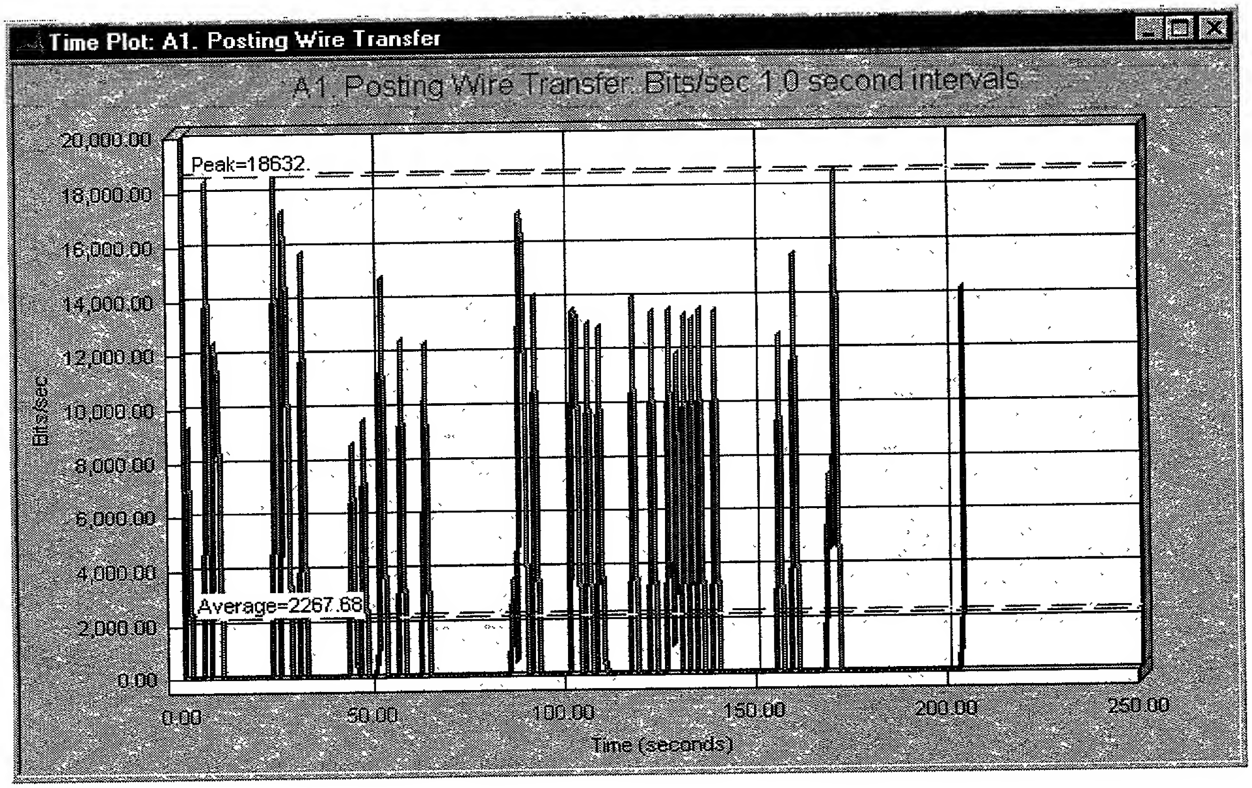


**Figure 7: Application Burstiness**

**Sensitivity**
Response time is the one of the most important aspects of an application. Regardless of whether an application bursts and consumes bandwidth or whether it communicates efficiently, the bottom line is that when users run the application they expect a certain level of performance. It is important to understand what network components have the greatest impact on an application's response time; this aspect is also called application sensitivity. The goal is to determine whether an application is more sensitive to varying amounts of bandwidth or network latency. The following diagram shows how response time is affected with varying amounts of bandwidth and network latency.

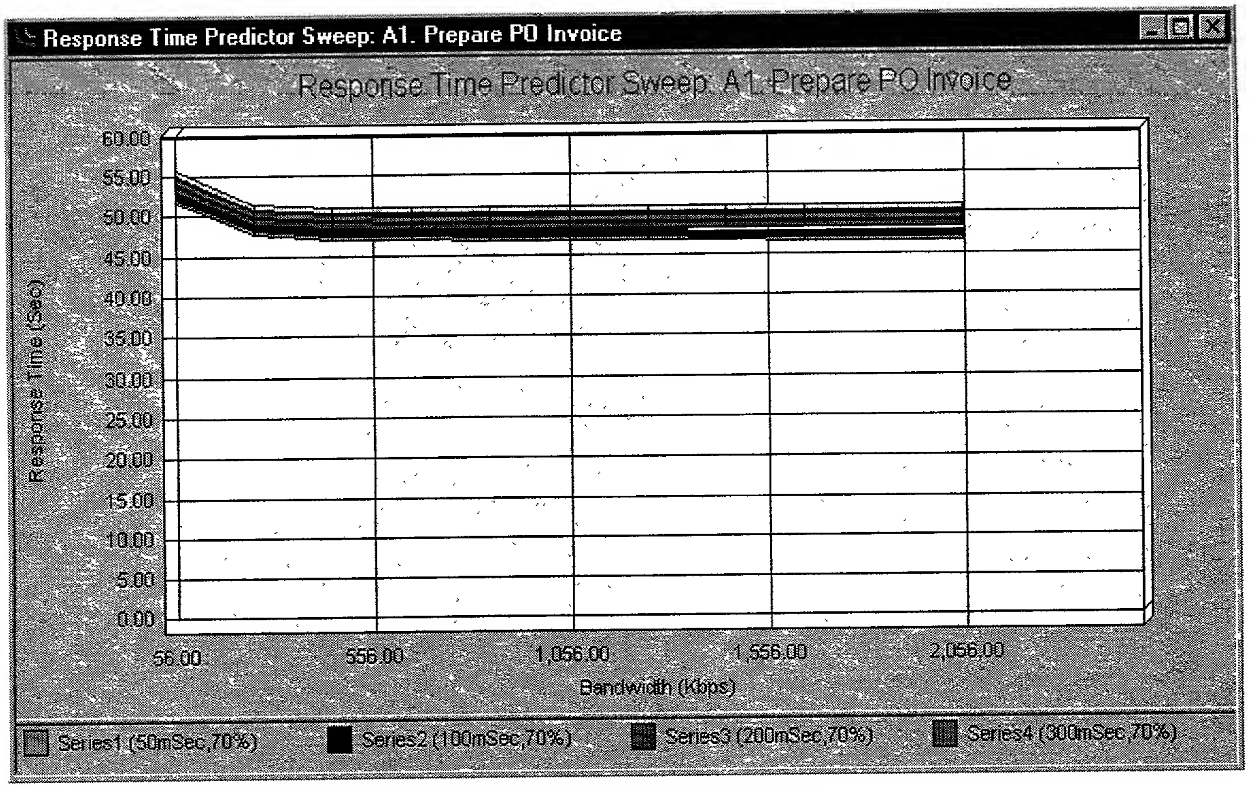


**Figure 8: Application Sensitivity**

The data on the ordinate (y-axis) represents the response time in seconds. The abscissa (x-axis) represents bandwidth in Kbps. Each of the "Series" at the bottom of the graph shows two numbers. The first is the amount of roundtrip latency that is present between two end-points and the second is the amount of load on the slowest link in the path. The key is to determine which aspect, bandwidth or latency, has a greater impact on the response time of a task. Although it may not appear evident from the above graph, the response time for this particular task is more sensitive to varying amounts of bandwidth. The actual calculation is done using the raw data from the graph. Every task is evaluated using this process. The overall observation is that R/3 has a tendency to be a more sensitive to bandwidth than varying amounts of network latency; response time will continue to improve as more bandwidth becomes available.

## Data Symmetry

Data symmetry is an important aspect to understand especially if the application is going to be implemented on any transport technology that supports asymmetric traffic flows, such as Frame Relay. By understanding the data symmetry (server to client ratio) a cost effective network solution can be designed that allocates the correct amount of bandwidth in either direction. Although the server to client ratio varies for many of the individual tasks, the overall ratio was observed to vary from 4:1 to 6:1, depending on the component that was tested.

## Sizing

One of the primary purposes of this document is to understand the bandwidth requirements for the users accessing the accounts payable and accounts receivable components of the Financials module. This will become an important issue when the SAP R/2 system in Europe is phased out and users start accessing the SAP R/3 instance in Lancaster.

The bandwidth requirements were determined separately for each component in order to help in the planning process as users start migrating over to the SAP R/3 system. It should be noted that all of the login components (as referenced by figure 6) were modeled as part of the simulation. The reason for doing this is to help ABC in the future make decisions on whether or not to re-distribute some of the components and how these changes might affect response time.

The following two diagrams are the results of the simulation based on a single user of the accounts payable and accounts receivable components.

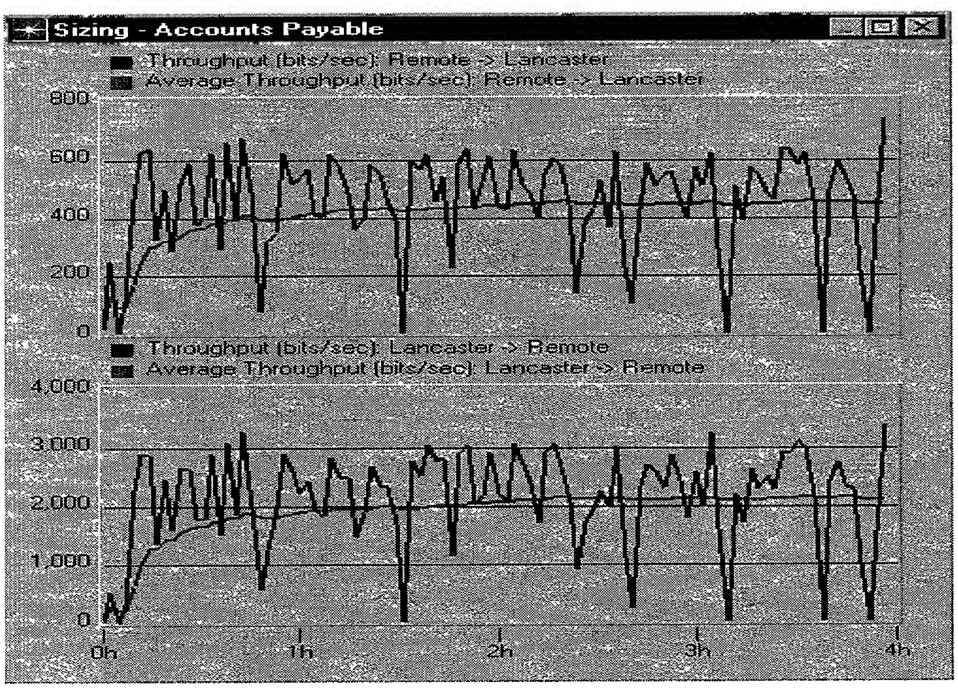


**Figure 9: Sizing - Accounts Payable**

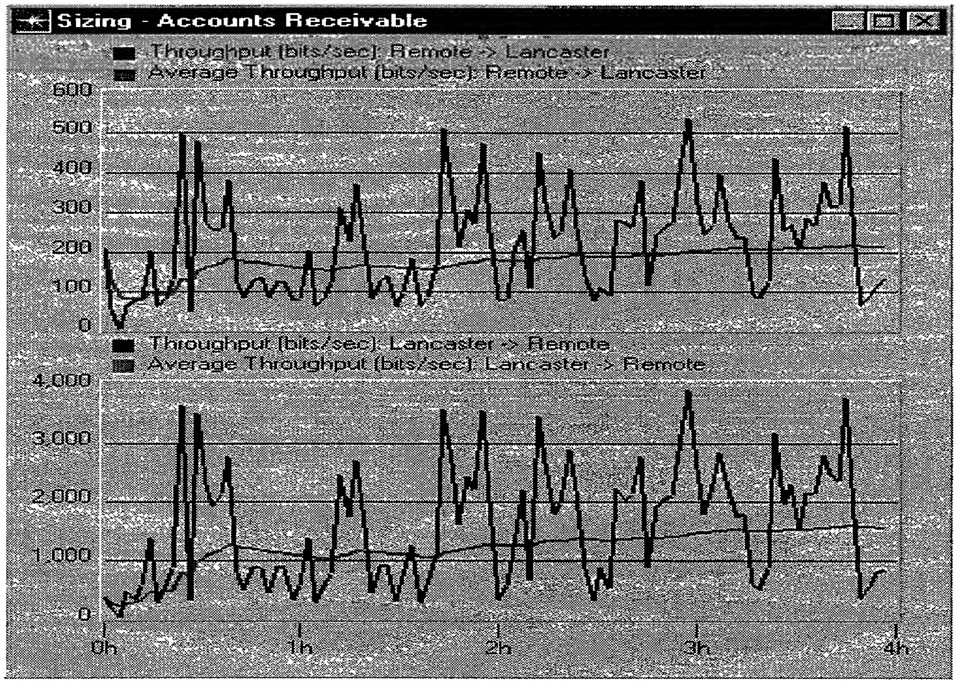


**Figure 10: Sizing - Accounts Receivable**

The numbers on the ordinate (y-axis) represent throughput in bits/sec and the numbers on the abscissa (x-axis) represent the length of the simulation in hours. Each simulation ran for a total of four hours in order to ensure that enough data points were generated and a trend could be observed.

Based on the test results the following is a summary of the recommended per user requirement for each of the tested components.

| | Client → Server | Server → Client |
|---|---|---|
| Accounts Payable | 500bps | 2Kbps |
| Accounts Receivable | 200bps | 1.2Kbps |

**Proposed Network Design**

Over the course of several days and representatives from each of ABC's three business units worked together in order to formulate a tentative network design moving forward based on several factors. The first was past experience and feedback from the user community in regards to network performance. The second was based on the anticipated application usage and distribution across the network.

The following six diagrams are the results of the network re-design and will be used as a basis for moving forward.

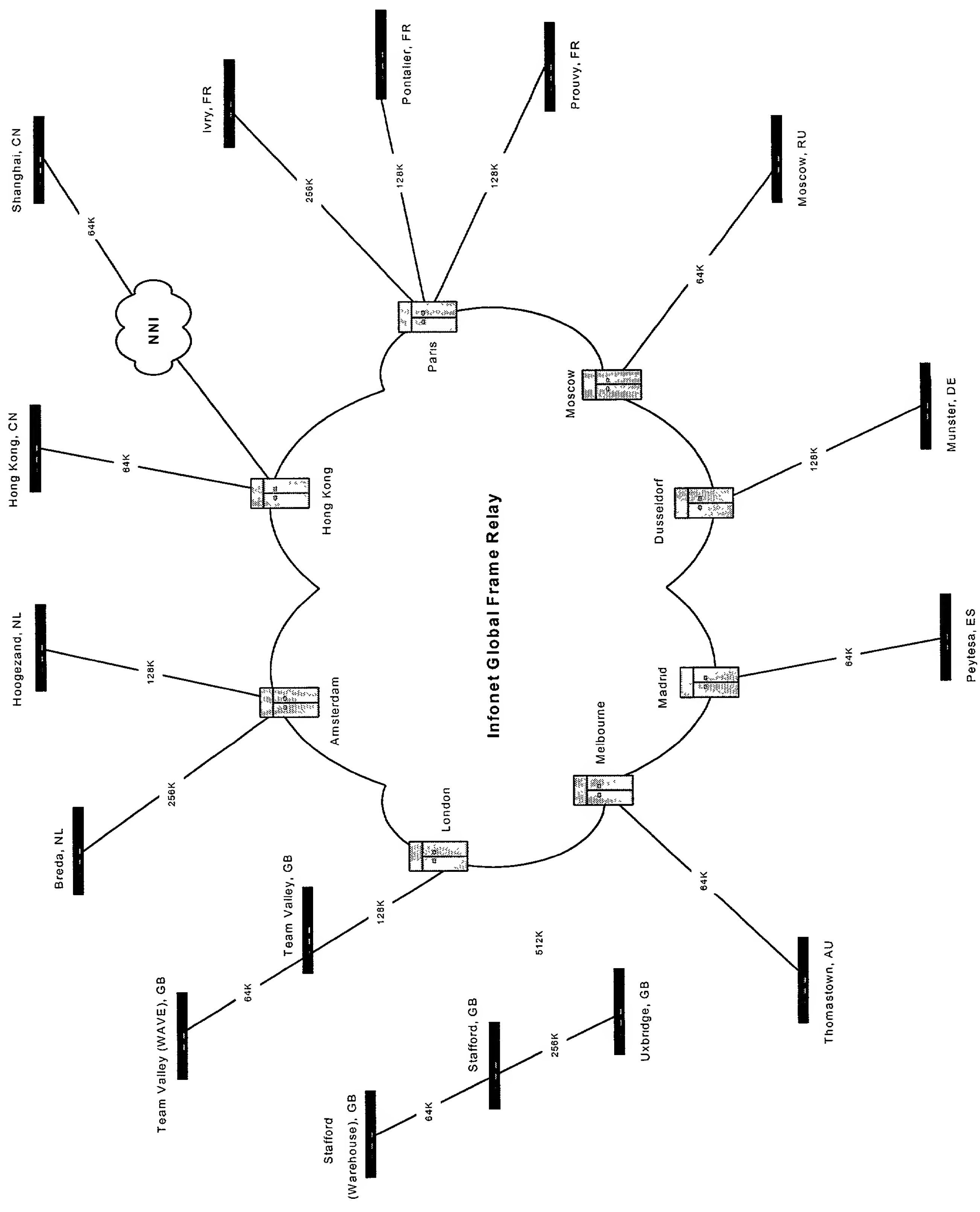


**Figure 11: BPO – Circuits**

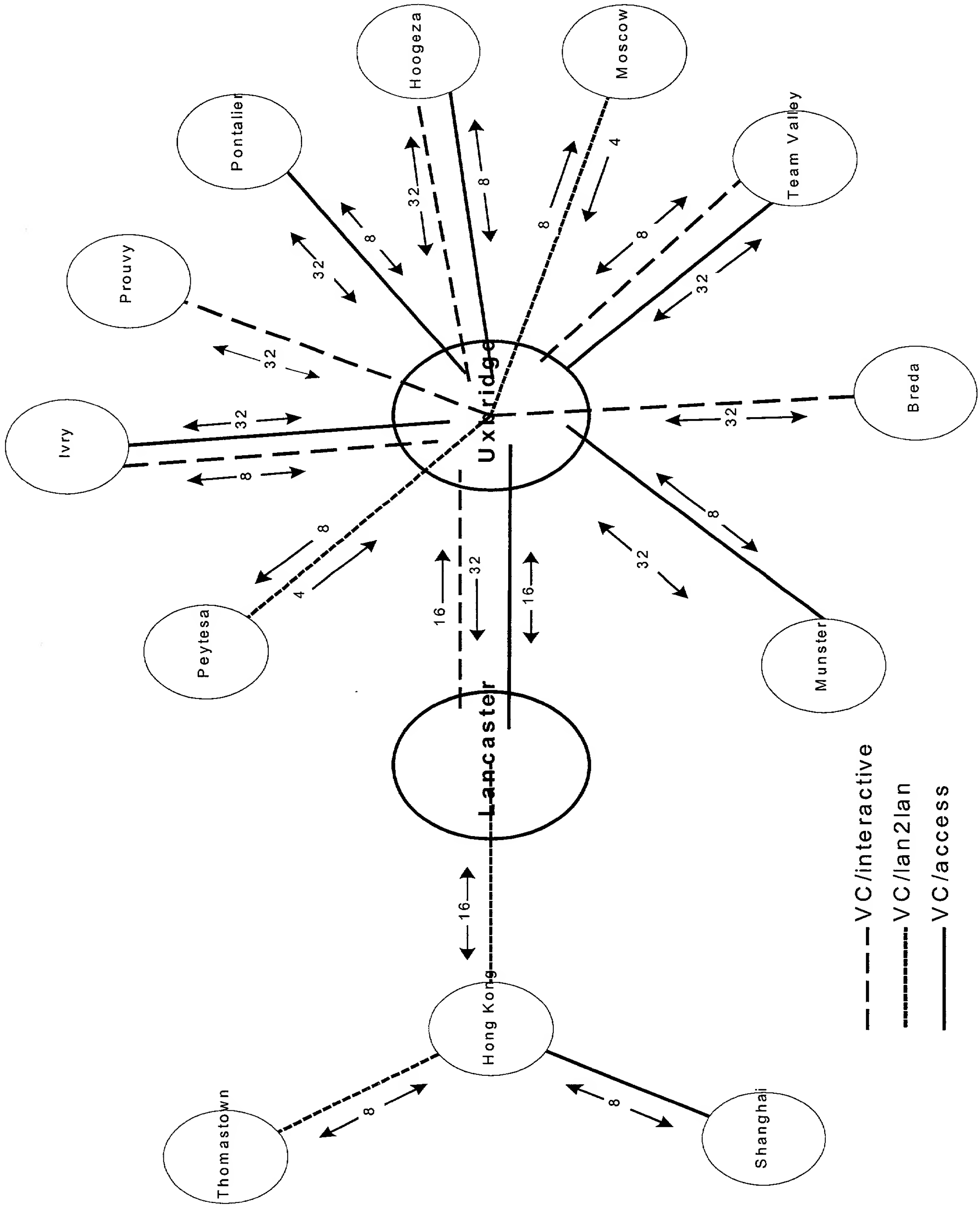


Figure 12: BPO – PVCs

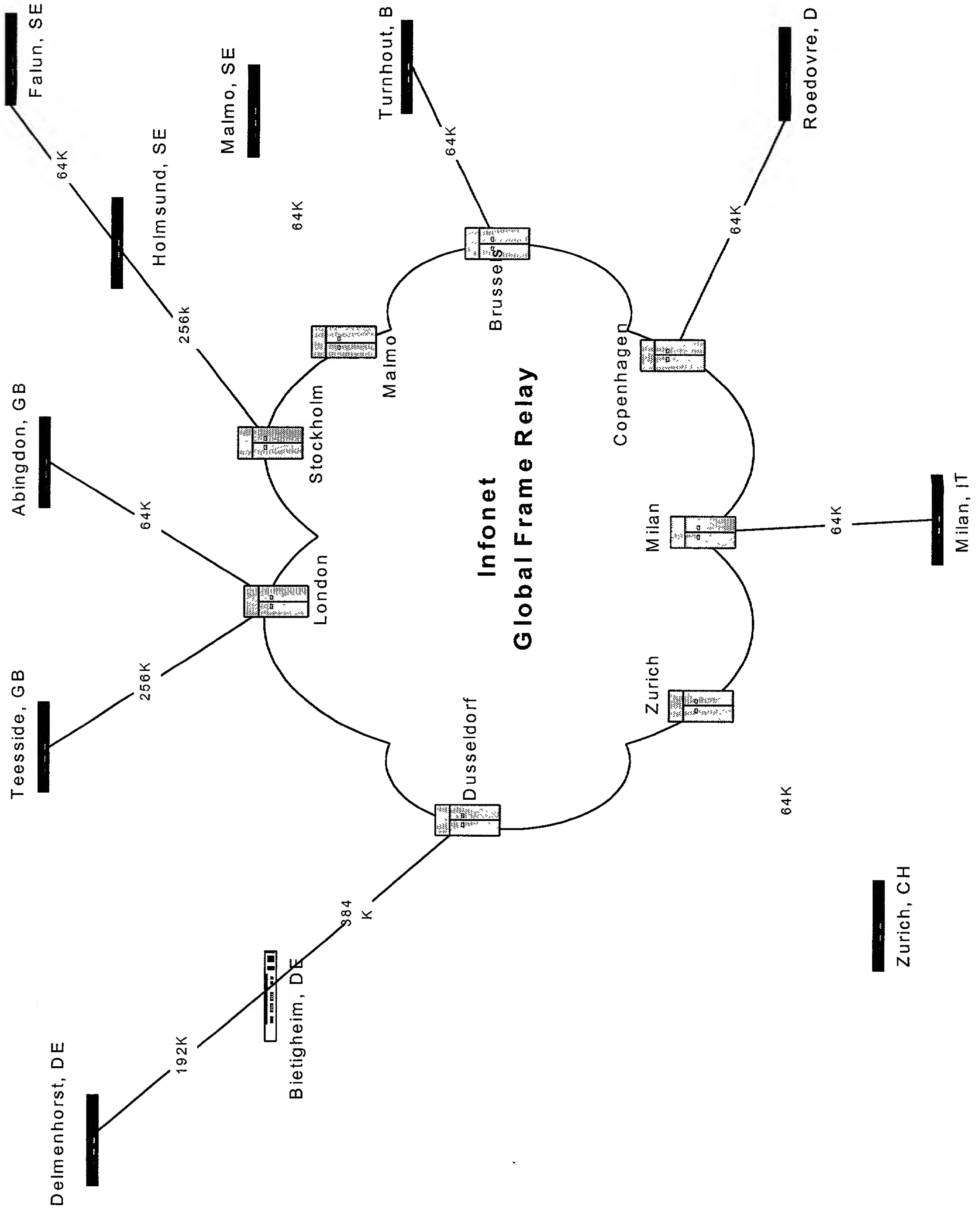


**Figure 13: ADE – Circuits**

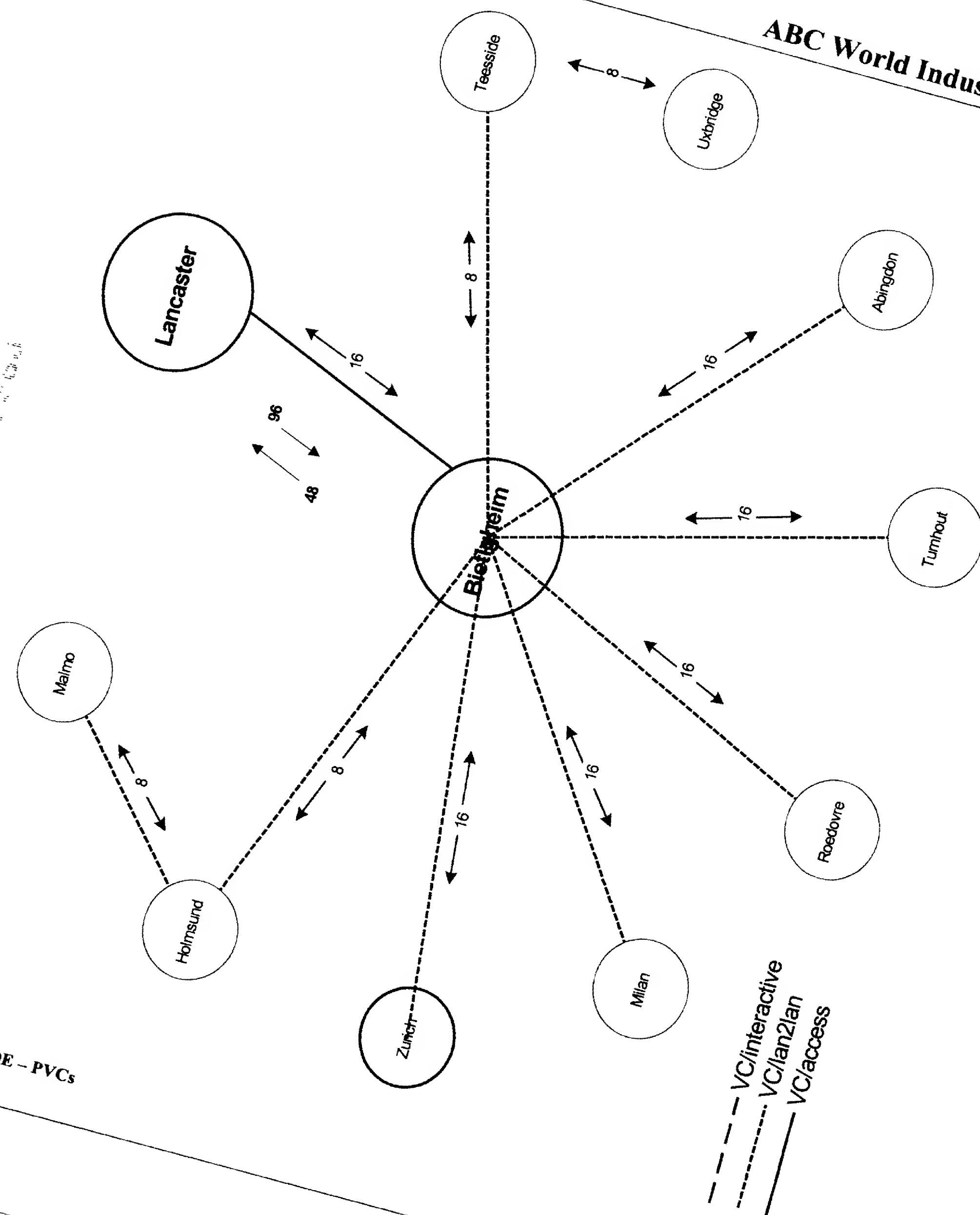


**Figure 14: ADE – PVCs**

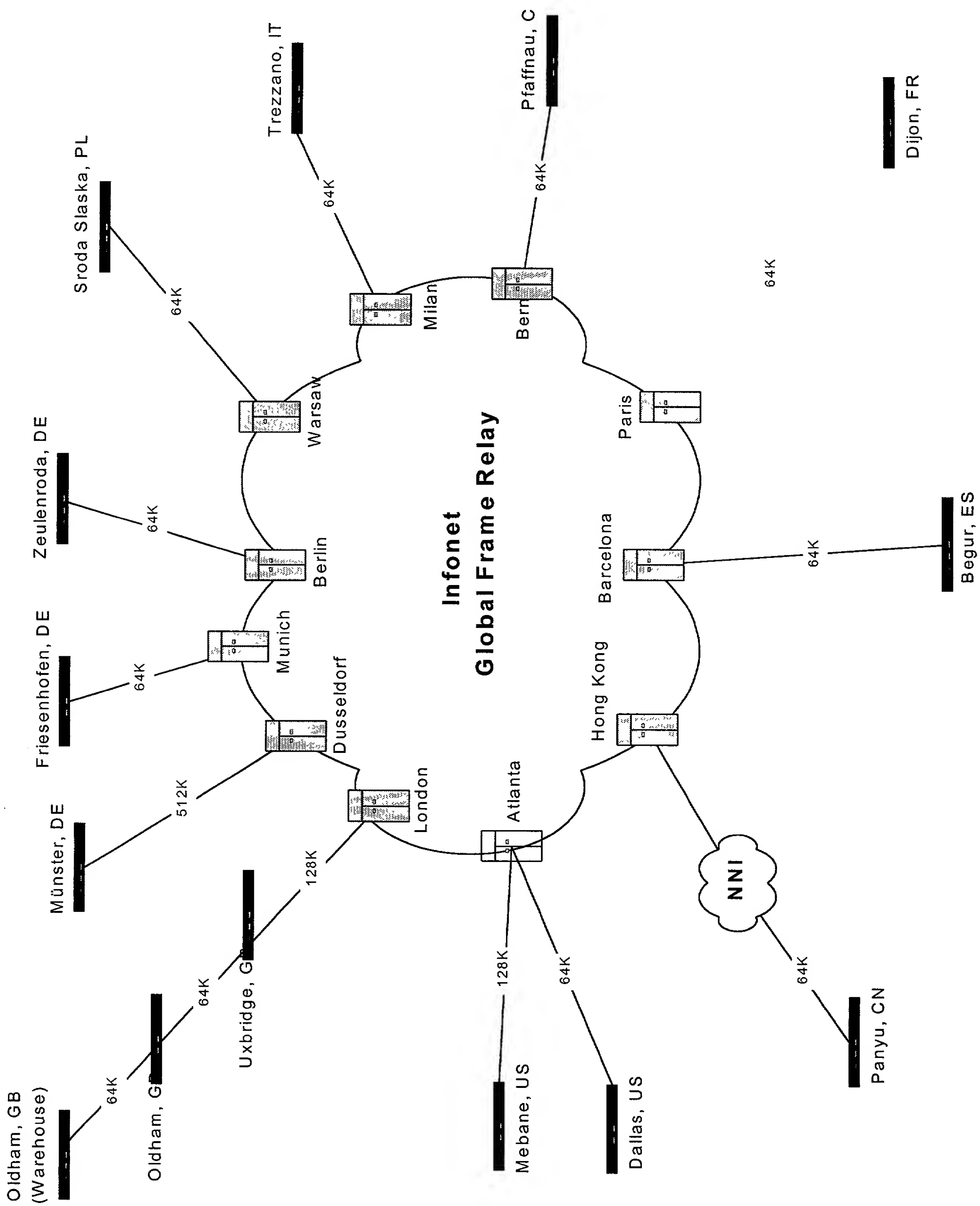


**Figure 15: XYZ – Circuits**

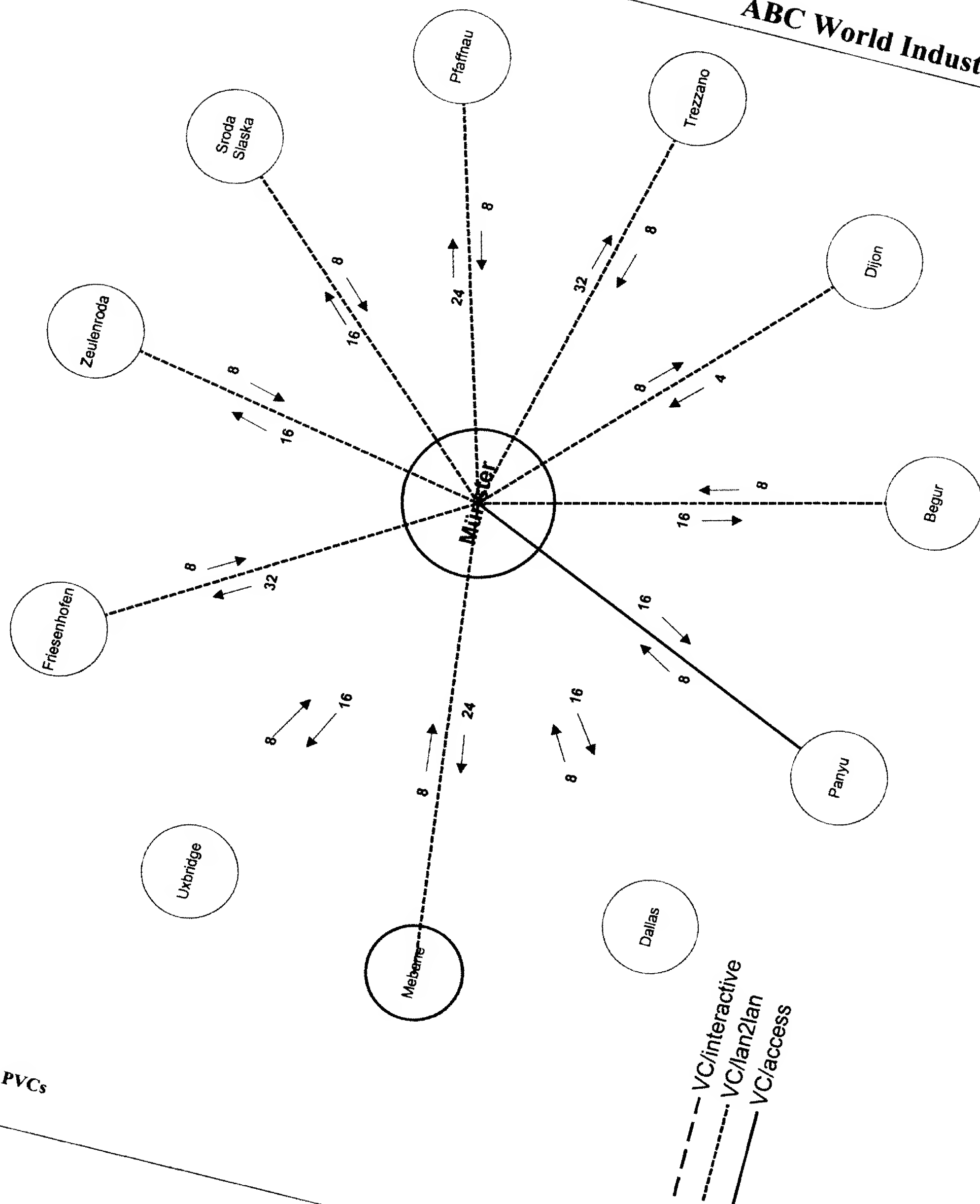


Figure 16: XYZ – PVCs

**Summary**

The purpose of this document was to profile the performance characteristics of two key components of the SAP R/3 Financials module and to establish a network re-design for the BPO, ADE, and XYZ.

Typically when completes a study it puts forth a series of recommendations that pertain to the issues that were addressed specifically during the analysis. However, this document represents a baseline, a snapshot in time, and should be used as a reference for moving forward as ABC continues to make changes as a result of application migrations and organizational changes dues to the changing complexion of ABC's organizational structure.

anticipates that over time ABC will provide updated information, which can be used as a source for modifying any of the existing components of the current network design. would like to work closely with ABC during this process. ABC should take advantage of the work that has been done, especially in the area of network modeling and simulation.